Quick Revision Note: Container With Most Water (LeetCode #11)

Concept:
We are given an array 'height' where each element represents the height of a vertical line at that index. We must choose two lines such that together with the x-axis they form a container, which holds the maximum water. **Approach:**
• Use **Two Pointer Technique**: one at the start (lp = 0) and one at the end (rp = n-1).
• Calculate current water: width × min(height[lp], height[rp]).
• Update maximum if current water is larger.
• Move the pointer pointing to the smaller height inward.
• Repeat until lp < rp.
Time Complexity: O(n)
Space Complexity: O(1)

```
class Solution {
    public int maxArea(int[] height) {
        int maxWater = 0;
        int lp = 0;
        int rp = height.length - 1;

        while(lp < rp) {
            int w = rp - lp;
            int ht = Math.min(height[lp], height[rp]);
            int currWater = w * ht;

            maxWater = Math.max(maxWater, currWater);
            if (height[lp] < height[rp]) {
                lp++;
            } else {
                rp--;
            }
        }
        return maxWater;
    }
}
```